Nombre: Esteban Pareja - **Correo**: tebanpar@gmail.com – Cel: 3207330571 (WhatsApp) –
Fecha: 14/11/2025

CONTENIDO

Introducción

El presente documento describe la propuesta de arquitectura de integración para la modernización de sistemas bancarios, desarrollada en el marco del ejercicio técnico solicitado. El objetivo principal es diseñar una solución que permita integrar los sistemas existentes y nuevos componentes tecnológicos, garantizando interoperabilidad, seguridad, alta disponibilidad y cumplimiento normativo, alineada con estándares de la industria como BIAN y prácticas modernas de arquitectura.

La solución propuesta aborda los siguientes aspectos clave:

- Diseño de arquitectura de alto nivel, representado mediante diagramas C4 (Contexto, Contenedores y Componentes).
- Selección y justificación de patrones de integración y tecnologías.
- Estrategias para seguridad, gobernanza de APIs y protección de datos.
- Mecanismos para alta disponibilidad, recuperación ante desastres y migración gradual.
- Preparación para Open Banking y Open Finance, mediante APIs seguras y gobernadas.

Arquitectura de alto nivel (C1, C2, C3).

Diagrama C1 - Contexto

El **Diagrama de Contexto** representa la interacción entre el **Core Bancario** y los sistemas internos y externos que conforman la arquitectura de integración. Este diseño sigue el estándar **BIAN**, asegurando interoperabilidad y alineación con dominios funcionales definidos por la industria.

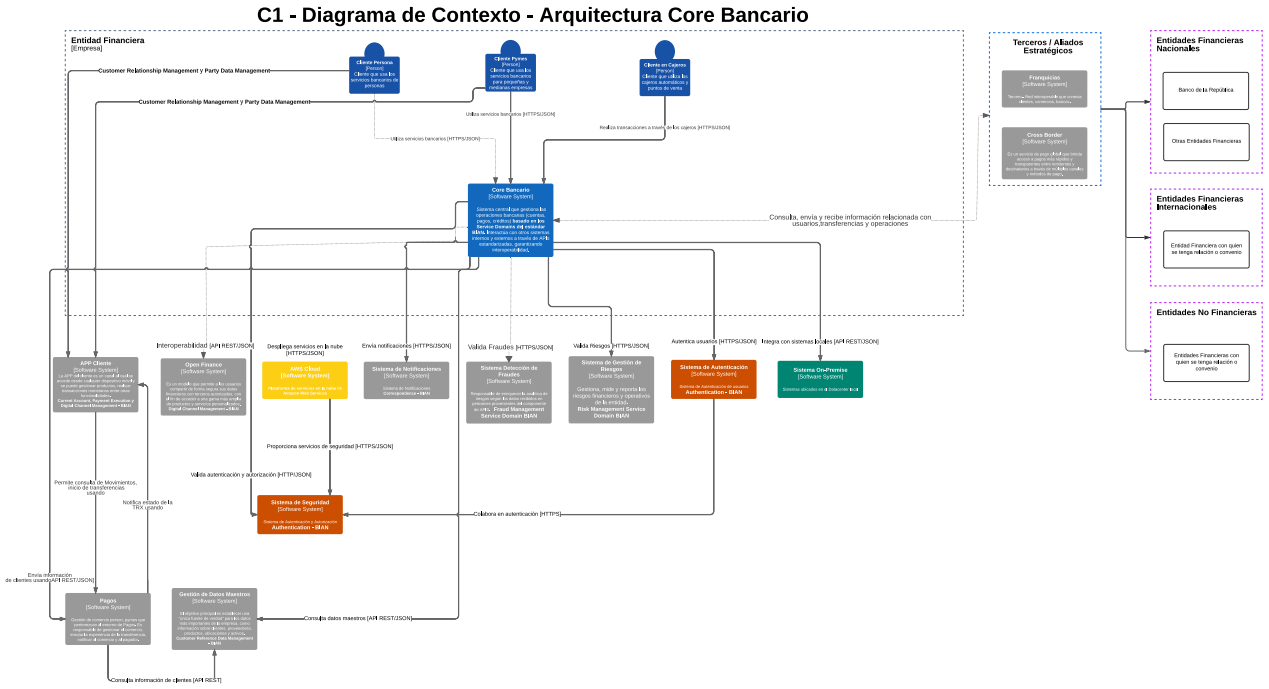


Diagrama C2 – Contenedores

Este nivel describe la arquitectura tecnológica que soporta la integración del **Core Bancario** y los servicios asociados.

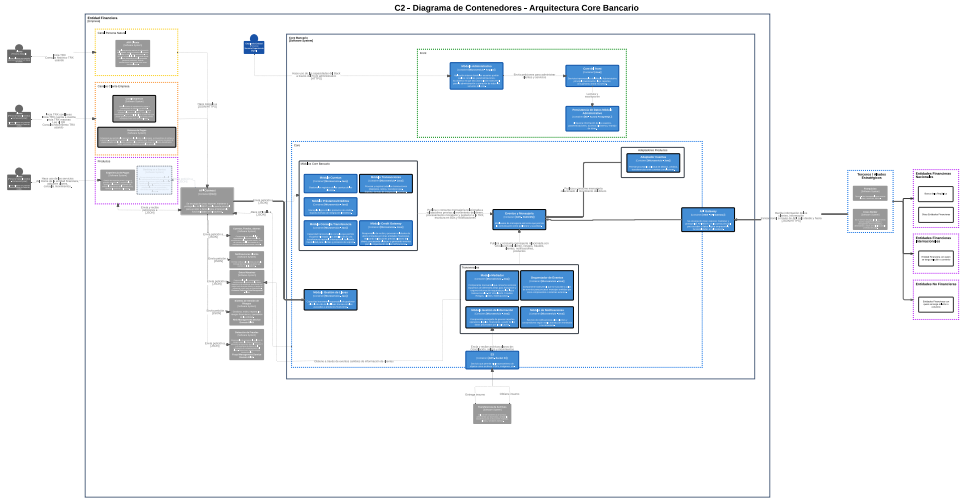
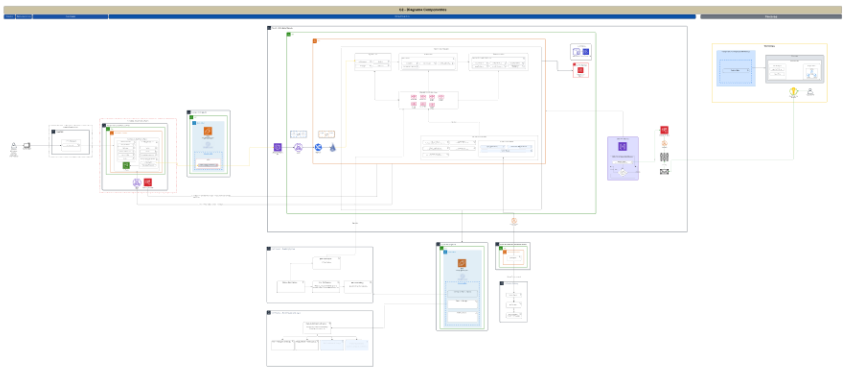


Diagrama C3 – Componentes

Este nivel permite entender la granularidad del diseño y cómo se cumplen los principios de



1. **Patrones de integración y tecnologías**.

La arquitectura propuesta aplica los siguientes patrones, cada uno justificado según su rol planteado en el diseño de las arquitecturas:

- **API Gateway**
  - **Dónde se aplica**: En el contenedor de integración principal.
  - **Por qué**: Centraliza la exposición de APIs internas y externas, aplicando políticas de seguridad, control de tráfico y versionamiento. Esto reduce la complejidad en los canales digitales y facilita la gobernanza.
- **Event-Driven Architecture (EDA)**
  - **Dónde se aplica**: En el Event Bus que conecta Core Bancario, Pagos, Riesgos y Fraudes.
  - **Por qué**: Permite integración desacoplada y resiliente. Si un servicio falla, los eventos se almacenan y se procesan cuando el sistema se recupera, garantizando continuidad operativa.
- **Microservicios**
  - **Dónde se aplica**: En los componentes del Core Digital, Pagos, Riesgos y Fraudes.
  - **Por qué**: Cada dominio funcional (BIAN) se implementa como un servicio independiente, lo que facilita escalabilidad, mantenimiento y despliegue incremental.
- **Identity Provider Centralizado**
  - **Dónde se aplica**: En el componente de autenticación y autorización.
  - **Por qué**: Garantiza seguridad y control de acceso uniforme en todos los sistemas, cumpliendo normativas de protección de datos.
- **API Composition**
  - **Dónde se aplica**: En el API Gateway para orquestar respuestas desde múltiples microservicios hacia los canales digitales.
  - **Por qué**: Mejora la experiencia del cliente al consolidar datos en una sola respuesta, reduciendo latencia.

2. **Requisitos de Seguridad, Cumplimiento Normativo y Protección de Datos**

La arquitectura propuesta incorpora medidas específicas para garantizar la seguridad y el cumplimiento regulatorio en cada capa del diseño:

- **Seguridad en la integración**
  - Autenticación y autorización centralizada mediante Identity Provider.
  - Protocolos OAuth2/OpenID Connect y tokens JWT para asegurar las llamadas internas y externas.
  - Cifrado extremo a extremo (HTTPS/TLS) en todas las comunicaciones.
  - Políticas en API Gateway: rate limiting, validación de tokens y control de acceso por roles.

- **Cumplimiento normativo**
  - **Ley de Protección de Datos Personales:**
  - Segmentación de datos sensibles en componentes dedicados (Gestión de Datos Maestros).
  - Acceso controlado y trazabilidad para auditorías.
  - **Regulaciones financieras (ej. PCI DSS, GDPR):**
  - Cifrado en reposo para información crítica.
  - Logs y monitoreo para garantizar trazabilidad y auditoría.

- **Protección de datos personales**
  - Principio de mínima exposición: Solo los datos necesarios viajan entre componentes.
  - Anonimización y enmascaramiento en procesos analíticos (riesgos y fraudes).
  - Gestión de consentimientos para Open Finance, asegurando que el cliente autoriza el uso de sus datos.

- **Observabilidad y respuesta ante incidentes**
  - Monitoreo centralizado (logs, métricas, alertas) para detectar anomalías.
  - Auditoría continua para garantizar cumplimiento normativo.
  - Planes de contingencia para incidentes de seguridad.

3. **Estrategia alta disponibilidad y recuperación ante desastres**.

La arquitectura propuesta garantiza alta disponibilidad (HA) y recuperación ante desastres (DR) mediante los siguientes mecanismos integrados en el diseño:

- **Despliegue híbrido**
  - Los componentes críticos (Core Bancario, API Gateway, EDA) se distribuyen entre infraestructura on-premise y servicios en la nube, reduciendo el riesgo de un único punto de falla.
- **Redundancia en contenedores y componentes**
  - Cada contenedor (Core, Pagos, Riesgos, Fraudes) se implementa con instancias replicadas en diferentes nodos, asegurando continuidad ante fallos.
- **EDA para resiliencia**

    - La mensajería asíncrona desacopla los sistemas, permitiendo que las operaciones continúen incluso si un servicio falla temporalmente.
    - Los eventos se almacenan en colas persistentes para garantizar la entrega.
- **Sincronización entre cores**
    - El Core tradicional y el Core digital se integran mediante APIs y eventos, lo que permite failover funcional: si uno se degrada, el otro puede asumir operaciones críticas.
- **Backups y restauración rápida**
    - Bases de datos del Core y sistemas asociados se respaldan periódicamente y se replican en entornos alternos (cloud/on-premise).
    - Procedimientos automatizados para restauración en caso de pérdida.
- **Monitoreo y alertas proactivas**
    - Observabilidad integrada (logs, métricas, trazas) para detectar incidentes y activar planes de contingencia.

4. **Propuesta estrategia de Integración multicore**.
    - **Gestión de identidad y acceso**.

    La arquitectura incluye:

    - Identity Provider centralizado para todos los sistemas.
    - Protocolos OAuth2/OpenID Connect para autenticación y autorización.
    - Control de roles y permisos aplicado en el API Gateway y en cada microservicio.
    - Tokens JWT para garantizar seguridad en las llamadas internas y externas.

5. Enfoque para la Gestión de Identidad y Acceso

La arquitectura propuesta garantiza la seguridad en la integración mediante un modelo centralizado de identidad y control de acceso, aplicado en todos los componentes:

1. Mecanismos propuestos

- **Identity Provider centralizado**: Gestiona autenticación y autorización para todos los sistemas.
- **Protocolos estándar**: OAuth2 y OpenID Connect para interoperabilidad segura.
- **Tokens JWT**: Firmados y con expiración para validar cada solicitud.

2. Métodos de autenticación

- **Autenticación basada en credenciales cifradas** (HTTPS/TLS).
- **Multi-factor Authentication (MFA)** para canales críticos (App y Web Banking).
- Integración con sistemas externos mediante **federación de identidad**.

3. Métodos de autorización

- **Control de acceso basado en roles (RBAC)**: Define permisos por tipo de usuario (cliente, administrador, tercero).
- **Políticas en API Gateway**: Validación de tokens, rate limiting y restricción por IP para APIs externas.
- **Microservicios**: Validación interna de permisos antes de ejecutar operaciones.

4. Aplicación en componentes del diseño

- **Canales digitales (App/Web)**: Autenticación inicial y emisión de token.
- **API Gateway**: Punto único para validar tokens y aplicar políticas.
- **Core Bancario y microservicios**: Autorización granular para operaciones críticas.
- **Open Finance**: Consentimiento explícito y validación de terceros autorizados.

6. **Estrategia de APIs internas y externas**.
   - **APIs internas**: Expuestas mediante API Gateway, con políticas de versionamiento y monitoreo.
   - **APIs externas**: Cumplen estándares de Open Banking/Open Finance, con autenticación segura y control de consumo.
   - **Gobierno**: Registro en catálogo central, auditoría y métricas para garantizar calidad y trazabilidad.

7. **Modelo de gobierno de APIs y microservicios**.
   - **Ciclo de vida controlado**: Publicación, pruebas, despliegue y retiros gestionados por API Manager.
   - **Monitoreo y observabilidad**: Logs, métricas y trazas para cada microservicio.
   - **Políticas de seguridad**: Validación de tokens, rate limiting y auditoría en todas las APIs.

8. **Plan de migración gradual**.
   - **Fase 1**: Exposición de APIs del core legacy mediante API Gateway.
   - **Fase 2**: Implementación de microservicios del nuevo core en paralelo, integrados por EDA.
   - **Fase 3**: Desactivación progresiva del core tradicional, manteniendo sincronización hasta completar la migración.
   - **Pruebas controladas** en cada fase para minimizar riesgos operativos.

Comentarios

URL Github

https://github.com/tebanpar-architecture/Devsu.git

Se adjuntan las imágenes de los diagramas C1, C2 y C3 en el repositorio de Git como en el de la prueba.